How Apache Solr Uses Apache Lucene

TONY MU | TONGM3@ILLINOIS.EDU | CS410 FALL 2021

Introduction

Google, as one of the most successful internet and software companies in the past two decades, has made a name out of its search engine and the number of Daily Active Users it has. Unfortunately, Google has not had in the past, nor will it have any plans in the future, to open source its search technologies. As a successful business, Google also will not open source its system architecture that supports nearly 100k searches per second, not without a premium at least. Those who want to implement their own custom search engines are not out of luck. Luckily, we have two great open-source projects that will suffice this requirement: Apache Solr and Apache Lucene.

Apache Solr is a popular search server built on top of Apache Lucene, while Apache Lucene is an open source, Java-based, information retrieval library. In this article, we will be examining in depths what each technology is, what their use cases are, and how one is built on top of another.

Technologies Deep Dive

Apache Solr

Apache Solr is a popular open-source search server designed to drive powerful document retrieval applications. Because Solr is based on open standards, it is also highly extensible. Solr queries are also simple HTTp request URLs and the response is a structured document that supports JSON, XML, CSV, or other formats. Solr is also scalable as it can handle large size of data as well as a high QPS.

Many scaling techniques may be used when scaling Solr to the next level. For example, if the document collection has grown beyond a single physical machine can handle, then this collection can be “sharded” into multiple collections, using a well-understood sharding algorithm. For example, for blog posts, the shard key can be the author’s name or genre.

Another scaling technique commonly used with Solr when dealing with high QPS is to increase so called the “Replication Factor”. What it basically does is add more replications of the collection, and thus increasing the

QPS capacity linearly with the number of machines (replicas). The draw back for this technique is it is difficult to maintain consistency between replicas, and care is needed when implementing a concurrent update.

Apache Lucene

Apache Lucene, on the other hand, is an open-source search software which releases a core search library, named Lucene core, as well as PyLucene. PyLucene is a python binding for Lucene. Apache Lucene offers powerful, accurate, and efficient search algorithms through a simple API. Lucene supports ranked searching, many powerful query types such as phrase queries, wildcard queries, proximity queries, range queries, and more. It is also scalable and offers high-performance indexing. Lucene can achieve over 150GB/hour on modern hardware, and only uses about 1MB heap on RAM. The storage requirements for the built index is also small, as it only requires about 20-30% the size of text indexed.

Difference

Comparing Apache Solr with Apache Lucene is like comparing apples to oranges. In other words, they are nothing alike. Indeed, Apache Solr is a search indexing server that can be configured and deployed across many hosting environments. On the other hand, Apache Lucene is a search indexing library that provides a set of APIs, which abstract away the complicated implementations on various tasks on building a search functionality, such as indexing, ranking, etc.

How Is Solr Built on Lucene

From a high level, Apache Solr internally uses Apache Lucene libraries to generate the indexes as well as to provide a user friendly search. To fully understand how Solr ultilizes Lucene at its core, we will need to do a deep-dive into the architecture of Solr.

The Solr project is best described in three layers. The first layer is a thin layer of client APIs, other interfaces such as Javascript, Python, Ruby, etc., as well as a SolrJ client.

The second layer is what's best described as a Solr Application layer. This layer contains most of the Solr magic such as the Solr Application Layer, which we can think of as a HTTP server that serves requests and returns response. There is also a component with Processing Units, which include De-duplication, Data Import, Language Detection, and Index Handler. The final component is where our interest lies: it is the Apache Lucene Core. This component includes an Index Searcher, Query Parser, Index Writer, Index Reader, and chain of text analysis (tokenizer and analyzers).

The third and final layer is a storage layer that stores metadata and indexes.

We will focus our discussion on the last component in the second layer (application layer) described above, namely the Apache Lucene Core. Apache Lucene core gets packages as library with the Apache Solr application. It provides core functionality for Solr such as index, query processing, searching data, ranking matched results, and returning them back.

Lucene's index falls into the family of indexes known as an inverted index. This is because it can list, for a term, the documents that contain it. Apache Solr (underlying Lucene) indexing is a specially designed data structure, stored in the filesystem as a set of index files.

Apache Lucene also comes with a variety of query implementations. Query parser is responsible for parsing the queries passed by the end search as a search string. Lucene provides TermQuery, BooleanQuery, PhraseQuery, PrefixQuery, RangeQuery, and so on.

Conclusion

To conclude, Apache Solr and Apache Lucene are completely different software. Apache Solr is a server, while Apache Lucene is a library. They do share a common theme on searching, and Apache Solr heavily relies on the functionalities that Apache Lucene implements and exposes through its client API. Lastly, they have different use cases. For example, if I am to build a highly scalable web search engine that can handle thousands of QPS and can search through TBs of documents, then Apache Solr is undoubtedly the best match, as it is open-source, out-of-box ready, and highly scalable. On the other hand, if I am to add search functionality inside a mobile app, or a windows application, then Apache Lucene will be the best match, as it is highly flexible and customizable, and provides a set of powerful APIs.

References

Google QPS https://www.internetlivestats.com/one-second/#google-band

A Quick Overview on Apache Solr. https://solr.apache.org/guide/8_10/a-quick-overview.html

Apache Lucene Core https://lucene.apache.org/core/index.html

Apache Solr Architecture https://www.oreilly.com/library/view/scaling-apache-solr/9781783981748/ch01s04.html